MULTIMEDIA UNIVERSITY

STUDENT ID NO

MULTIMEDIA UNIVERSITY

FINAL EXAMINATION

TRIMESTER 2, 2018 / 2019 SESSION

DPJ 5018 – PROGRAMMING IN JAVA

(DIT Only)

11 MARCH 2019
9.00 a.m. – 11.00 a.m.
(2 Hours)

INSTRUCTIONS TO STUDENT

1. This question paper consists of 10 pages.
2. There are **TWO (2)** sections:
   - **Section A:** Three (3) questions. Each question carries 10 marks. Answer ALL questions.
   - **Section B:** Two (2) questions. Each question carries 20 marks. **Choose ONE (1) question only.**
3. Total mark is **50** marks.
4. Please write your answers in the **Answer Booklet** provided.

SECTION A: STRUCTURED QUESTIONS (30 MARKS)

Answer ALL questions. Each question carries 10 marks.

QUESTION 1 [10 MARKS]

Refer to the following two-dimensional array declaration.

```
int [][] votes = {{25,24,37,27}, {20,35,23,40},
                  {43,20,30,32}};
```

The array shows the numbers of votes received by the candidates from four clubs as shown in Table 1.

Candidate/Club	Chess	Archery	Badminton	Music
Anna	25	24	37	27
Kristy	20	35	23	40
Robin	43	20	30	32

Table 1: Number of votes

Write a method named `getTotalVotes()` that accepts the array as the argument and displays the total votes for each candidate. This method is not returning any value. You may refer the following output: (10 Marks)

```
run:
Candidate 1: Total votes:113
Candidate 2: Total votes:118
Candidate 3: Total votes:125
```

[TOTAL 10 MARKS]

Continued...

QUESTION 2 [10 MARKS]

a) Refer to the following class declaration and answer the following questions.
(5 Marks)

```
public class Course
{
    private String courseID, courseName;
    private double fee;

    public Course()
    {}

    public String getCourseID()
    {
        return courseID;
    }
}
```

i. Write another constructor for `Course` class that accepts course ID, course name and fee as the parameters and assigns them to the appropriate members. *[2.5m]*

ii. Write a mutator method for the `fee` field. *[2.5m]*

Continued...

b) What is the output of ***Program 1*** (***CutePet.java***)? (5 Marks)

```
class Cat
{   public Cat(){
        System.out.println("Cats are adorable");
    }
    public Cat(String a)
    {
        this();
        System.out.println("Its name is "+a);
    }
}
class Momo extends Cat
{
    public Momo()
    {
        super();
        System.out.println("Momo loves to eat salmon");
    }
}

class Anycats extends Cat
{
    public Anycats()
    {
        System.out.println("It is just a random cat");
    }
    public Anycats(String c)
    {
        super(c);
        System.out.println("It loves to sleep on my
laptop");
    }
}
public class CutePet
{
    public static void main(String[] args)
    {
        Momo obj = new Momo();
        Anycats obj2 = new Anycats("Toothless");
     }
}
```

Program 1: CutePet.java

[TOTAL 10 MARKS]

Continued...

QUESTION 3 [10 MARKS]

a) Refer to ***Program 2 (TrialTest.java)*** for the following questions. Write the output of the program if user enters: (5 Marks)

i. *3* for `text`. *[2.5m]*

ii. ***Three*** for `text`. *[2.5m]*

```
import java.util.Scanner;
public class TrialTest
{
    public static void main (String [] arg)
    {
        String text;
        try{
            text = new Scanner(System.in).next();
            int value = Integer.parseInt(text);

            int [] array = new int[value];

            for(int i=0;i<=value;i++)
            {
                array[i]=i*2;
                System.out.println(array[i]);
            }
        }
        catch(ArrayIndexOutOfBoundsException e)
        {
            System.out.println("Out of bounds.");
        }
        catch(IllegalArgumentException e)
        {
            System.out.println("Bad format.");
        }
        catch(RuntimeException e)
        {
            System.out.println("Not a number.");
        }
        finally{
            System.out.println("Unsuccessful run.");
        }
        System.out.println("Thank you!");
    }
}
```

Program 2: TrialTest.java

Continued...

b) Write the code for the following questions: (5 Marks)

i. Create a checkbox button with the text 'Female'. *[1m]*

ii. Create a drop-down button with these texts: FIST, FCI. *[1.5m]*

iii. Set a panel with right-aligned of `FlowLayout`. *[1.5m]*

iv. Create a font with the specifications: style – bold, size – 12, font name – Calibri. *[1m]*

[TOTAL 10 MARKS]

SECTION B: APPLICATION QUESTION (20 MARKS)

Choose ONE question only.

QUESTION 1 [20 MARKS]

In a company, a contract staff is entitled 10 days for annual leave. However, he/she is granted for additional leave (for next year) based on the total hours of overtime he/she has worked. Refer to the UML class diagrams as shown in ***Figure Q1a*** for the following instructions:

a) Write a class named `ContractEmployee` that extends the `Employee` class. The class consists of the following fields, constructor and methods.

- The `totalovertime` field represents the total hours of overtime that the contract staff has worked.
- The `takenLeave` field represents the total days of leave taken by the contract staff.
- The `grantedleave` field represents the total days of additional leave granted to the staff for next year.
- The `CONTRACTLEAVE` is a constant field of annual leave for contract staff with the value of 10.
- Constructor `ContractEmployee()` constructs an object of `ContractEmployee` by calling the superclass's constructor with parameters (as shown in the class diagram) and set the data accordingly.
- Method `getTotalOvertime` sums up the number of total overtime that the employee has worked for the past 12 months. This method accepts an `int` array as the parameter.
- Method `getGrantedLeave` determines the additional leave granted for this staff based on the total hours of overtime. Refer to the following table (***Table 2***):

Continued...

Total hours of overtime	Additional leave (day)
80 and above	6
60 - 79	5
40 - 59	4
30 - 39	2
24 - 29	1
Less than 24	No leave

Table 2: Additional leave for overtime

- Method `getCarryForwardLeave()` calculates the number of leave to be carry-forwarded for next year with the formula:

Carry-forward leave = contract staff annual leave – number of taken leave + additional leave granted.

b) Write an application program to test the `ContractEmployee` class.
- Prompt the user to key-in employee's name, ID and number of leave taken for this year.
- Create a `ContractEmployee` object with the parameters of the staff name, ID and the leave.
- Then, create an array of `int` type size 12 to store the overtime hours.
- Prompt the user to key-in the overtime hours for each month, and store in the array accordingly.
- Display all the details by calling the appropriate methods: (Refer to ***Figure Q1b***)
  - The employee's name, ID and leave taken.
  - Total number of overtime.
  - Total number of granted leave.
  - Total number of carry-forward leave for next year.

(Note: You DO NOT have to write the `Employee` class)

Continued...

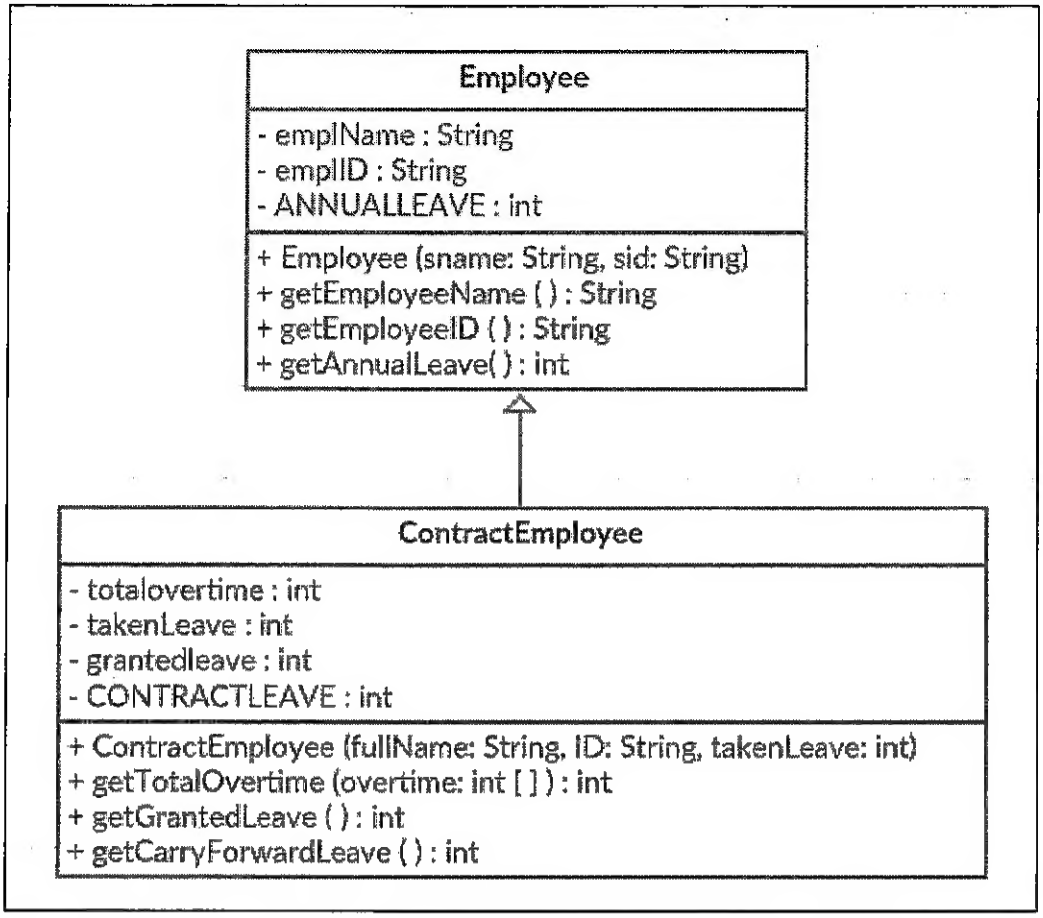

Figure Q1a: UML class diagrams for Employee and ContractEmployee

Continued...

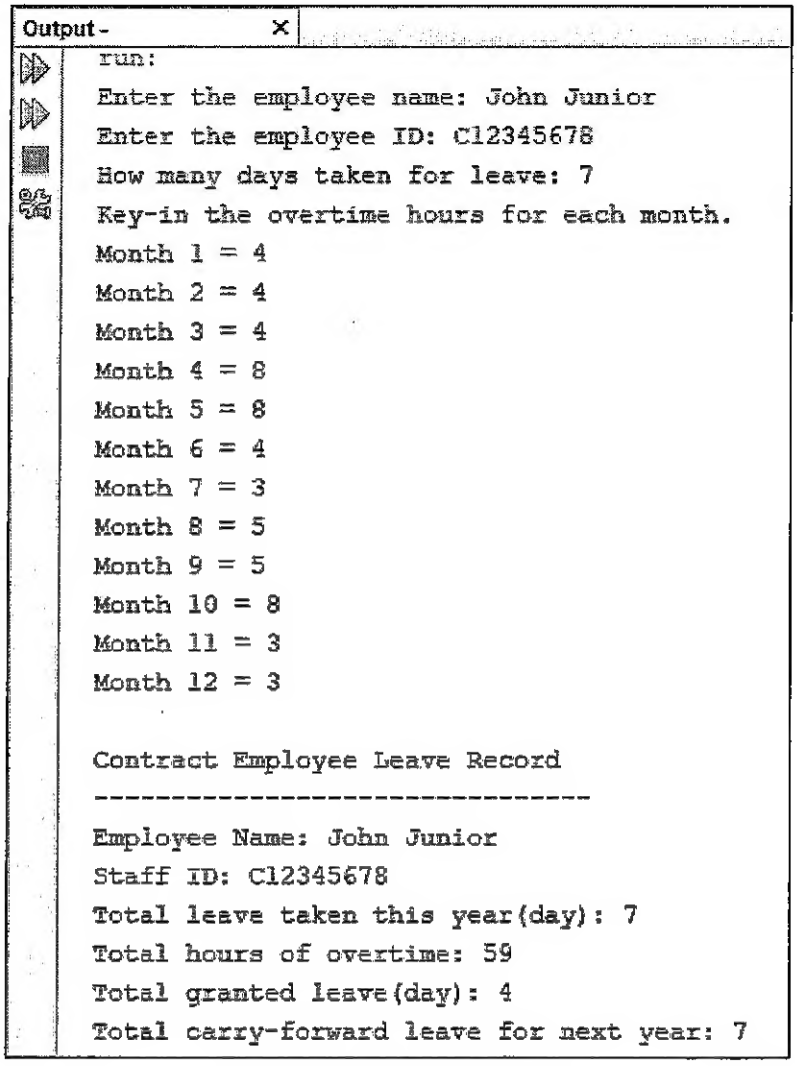

Figure Q1b: Output screen

[TOTAL 20 MARKS]

Continued...

QUESTION 2 [20 MARKS]

Write an **Applet** program that calculates the total cost of services by Mommy Folding and Ironing Service. User chooses the services (folding or ironing or both) from check boxes and enters the details of the services accordingly – weight (kg) for the folding clothes and number of pieces of clothes for the ironing service. The rate for folding is RM 5.00 per kg and rate for ironing is RM 1.00 per piece. Delivery charge is RM 10.00 if needed.

Once user clicks the button, the costs are displayed in the text fields respectively.

Refer to the ***Figure Q2a*** for the layout and ***Q2b***, ***Q2c*** for the output screens.

(Note: You DO NOT have to write the HTML file)

<table>
<tr><td rowspan="2">Panel 1</td><td colspan="3">Label with icon ("icon.png"), blue colour
font : Sanserif, size 18, bold and italic</td></tr>
<tr><td>Label, check box 1, check box 2
(Panel 2)</td><td>Label
Text field
Label
Text field
(Panel 3)</td><td>Radio button and Button
(Panel 4)</td></tr>
<tr><td rowspan="3">Panel 5</td><td colspan="2">Label</td><td>Text field</td></tr>
<tr><td colspan="2">Label</td><td>Text field</td></tr>
<tr><td colspan="2">Label</td><td>Text field</td></tr>
</table>

Figure Q2a: The Applet layout

- Note:
  - Panel 1 is located at the *NORTH* of `BorderLayout`
  - Panel 2 is located at the *EAST* of `BorderLayout`
  - Panel 3 is located at the *CENTER* of `BorderLayout`
  - Panel 4 is located at the *WEST* of `BorderLayout`
  - Panel 5 with `GridLayout(3,2)` is located at the *SOUTH* of `BorderLayout`.

Continued...

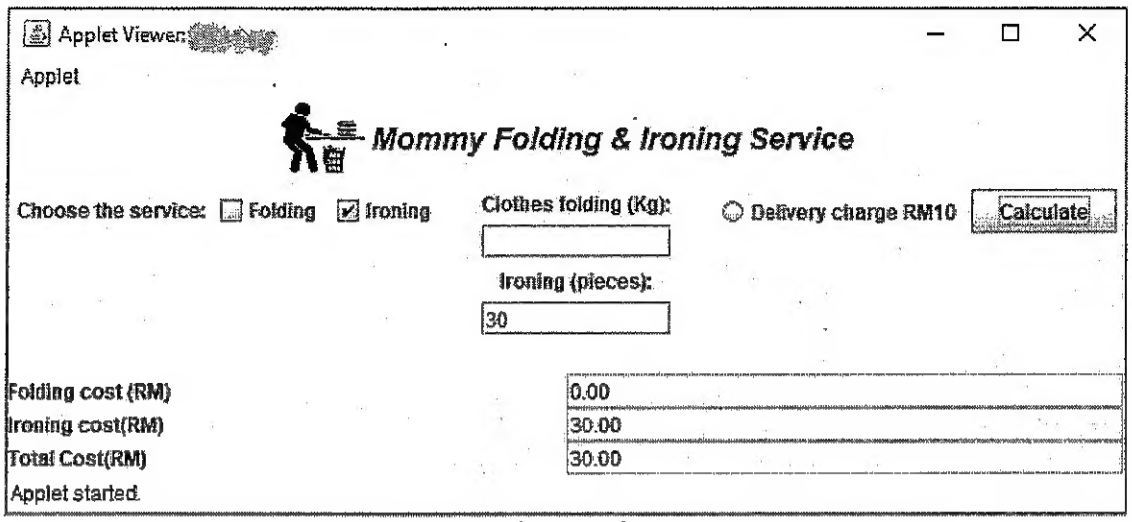

Figure Q2b: Sample output 1

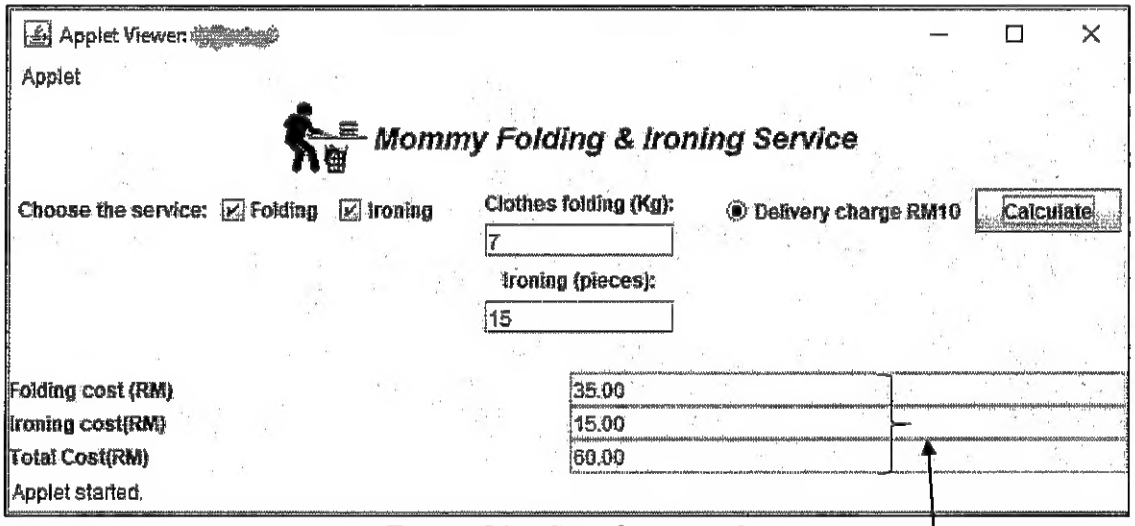

Figure Q2c: Sample output 2

These three text fields are non-editable.

[TOTAL 20 MARKS]

End of Page.